CLAIMS

What is claimed is:

1. A processor that includes a mechanism for detecting soft errors comprising:

a) instruction fetch unit for fetching an instruction;

b) an instruction decoder for decoding the instruction;

c) duplication hardware for duplicating the instruction;

d) a first execution unit for executing the instruction in a first execution cycle;

e) the first execution unit executing the duplicated instruction in a second execution cycle;

f) comparison hardware for comparing the results of the first execution cycle and the results of the second execution cycle; and

g) a commit unit for committing one of the results when the results are the same; and

h) an exception unit for generating an exception (raising a fault) when the results are not the same.

2. The processor of claim 1

wherein the step of executing the instruction in a first execution cycle includes the step of storing the results of the first execution cycle.

3. The processor of claim 1

wherein the step of executing the instruction in the first execution cycle includes issuing the decoded instruction to a first execution unit; and

wherein the step of executing the instruction in the second execution cycle includes issuing the decoded instruction to the first execution unit.

4. The processor of claim 3

wherein the first execution unit is one of floating point unit, an integer unit, a arithmetic logic unit (ALU), a multimedia unit, and a branch unit.

5. The processor of claim 1 further comprising:

a control register that includes a bit for enabling the duplication hardware and comparison hardware.

6. The processor of claim 5 wherein the bit is set by one of user-programmed firmware and an operating system.

7. The processor of claim 1 wherein each instruction includes a bit for enabling the instruction for error checking.

8. A method for detecting errors in a processor comprising the steps of:

a) fetching an instruction;

b) decoding the instruction;

c) duplicating the instruction;

d) executing the instruction in a first execution cycle;

e) executing the duplicated instruction in a second execution cycle;

f) comparing the results of the first execution cycle and the results of the second execution cycle; and

g) when the results are the same, committing one of the results; and

h) when the results are not the same, raising a fault.

9. The method of claim 8

wherein the step of executing the instruction in a first execution cycle includes the step of storing the results of the first execution cycle.

10. The method of claim 8

wherein the step of executing the instruction in the first execution cycle includes issuing the decoded instruction to a first execution unit; and

wherein the step of executing the instruction in the second execution cycle includes issuing the decoded instruction to the first execution unit.

11. The method of claim 8

wherein the execution unit is one of floating point unit, an integer unit, an arithmetic logic unit (ALU), a multimedia unit, and a branch unit.

12. The method of claim 8 wherein duplication hardware is provided for performing the instruction duplication and comparison hardware is provided for performing the comparison, the method further comprising the step of:

setting a bit in a control register;

wherein the bit enables the duplication hardware and comparison hardware.

13. The method of claim 12 wherein the bit is set by one of user-programmed firmware and an operating system.

14. A method for selectively enabling an error detection mechanism that employs alternating threads, comprising the steps of:

a) maintaining a control register that includes an error detection enable bit;

b) setting the error detection enable bit to enable the error detection mechanism; and

c) clearing the error detection enable bit to disable the error detection mechanism.

15. The method of claim 14 wherein the step of setting the error detection enable bit to enable the error detection mechanism includes one of

a user-programmed firmware setting the error detection enable bit to enable the error detection mechanism;

an operating system setting the error detection enable bit to enable the error detection mechanism; and

an application setting the error detection enable bit to enable the error detection mechanism; and

wherein the step of clearing the error detection enable bit to disable the error detection mechanism includes one of

a user-programmed firmware clearing the error detection enable bit to enable the error detection mechanism;

an operating system setting clearing the error detection enable bit to enable the error detection mechanism; and

an application clearing the error detection enable bit to enable the error detection mechanism.

16. The method of claim 14 wherein the error detection mechanism is enabled for a portion of critical code that includes a first instruction and a last instruction;

wherein the step of setting the error detection enable bit to enable the error detection mechanism includes the step of

setting the error detection enable bit to enable the error detection mechanism prior to the execution of the first instruction of the critical portion of code; and

wherein clearing the error detection enable bit to disable the error detection mechanism includes

clearing the error detection enable bit to disable the error detection mechanism after the execution of the last instruction of the critical portion of code.

17. An apparatus for executing instructions comprising:

a) a control register that includes an error detection enable bit;

b) an error detection mechanism for detecting soft errors; and

c) a mechanism for selectively enabling the error detection mechanism by setting the error detection enable bit to enable the error detection mechanism and by clearing the error detection enable bit to disable the error detection mechanism; wherein the error detection mechanism employs alternating threads.

18. The apparatus of claim 17 wherein the selective enabling mechanism is one of a user-programmed firmware, an operating system, and an application.

19. The apparatus of claim 17 wherein the error detection mechanism is enabled for a portion of critical code that includes a first instruction and a last instruction;

wherein the selective enabling mechanism sets the error detection enable bit to enable the error detection mechanism prior to the execution of the first instruction of the critical portion of code; and

wherein the selective enabling mechanism clears the error detection enable bit to disable the error detection mechanism after the execution of the last instruction of the critical portion of code.